

# A two-dimensional bisection method using the Poincaré-Miranda Theorem

Manuel López Galván\*

## Abstract

In this paper we develop a root-finding algorithm in order to solve non-linear systems of two equations using the Poincaré-Miranda theorem. This theorem is the natural extension of the Bolzano's theorem for the multidimensional case and it looks for sign conditions throughout the border domain in order to guarantee the existence of a root. In this work we propose a two-dimensional bisection method following the terms of this theorem. Given a system to solve and given a starting rectangle verifying the border conditions, we perform refinement of the rectangle following the sign conditions and taking the center we generate root's approximation for the system.

## 1 Introduction

The problem of finding numerical approximations to the roots of a non-linear system of equations was subject of various studies, different methodologies have been proposed between optimization and Newton's procedures. In [2] D. H. Lehmer proposed a method for solving polynomial equations in the complex plane testing increasingly smaller disks for the presence or absence of roots. In other work, Herbert S. Wilf developed a global root finder of polynomials of one complex variable inside any rectangular region using Sturm sequences[7].

The classical Bolzano's theorem or Intermediate Value theorem ensure that a continuous function that changes sign in an interval has a root, that is, if  $f : [a, b] \rightarrow \mathbb{R}$  is continuous and  $f(a)f(b) < 0$  then there exist  $c \in (a, b)$  that  $f(c) = 0$ . In the multidimensional case the generalization of this result is the

---

\*Supported by Instituto Argentino de Matemática (CONICET) y Universidad de Buenos Aires.

known Poincaré-Miranda theorem that ensure that if we have  $f_1, \dots, f_n$   $n$ -continuous functions of  $n$  variables  $x_1, \dots, x_n$  and the variables are subjected to vary between  $a_i$  and  $-a_i$  then if  $f_i(x_1, \dots, a_i, \dots, x_n) f_i(x_1, \dots, -a_i, \dots, x_n) < 0$  for all  $x_i$  then there exist  $c \in [-a_i, a_i]^n$  such that  $f(c) = 0$ . This result was announced the first time by Poincaré in 1883 [4] and published in 1884 [5] with reference to a proof using homotopy invariance of the index. The result obtained by Poincaré has come to be known as the theorem of Miranda, who in 1940 showed that it is equivalent to the Brouwer fixed point [3]. For different proofs of the Poincaré-Miranda theorem in the  $N$ -dimensional case, see [1], [6]. In the two-dimensional case the Poincaré-Miranda theorem can be enunciated in the following way.

**Theorem 1.1.** (*Poincaré-Miranda theorem*). *Let  $(f_1, f_2) : R = [-a_1, a_1] \times [-a_2, a_2] \rightarrow \mathbb{R}^2$  be a continuous vector field such that  $f_1(-a_1, y) f_1(a_1, y) \leq 0$ , for each  $|y| \leq a_2$  and  $f_2(x, -a_2) f_2(x, a_2) \leq 0$ , for each  $|x| \leq a_1$ . Then there exist  $(r_1, r_2) \in R$  such that  $f_1(r_1, r_2) = f_2(r_1, r_2) = 0$ .*

It is clear that the above theorem is still true assuming a non-symmetric square  $R$ . Although this paper  $\mathbf{F}$  will denote a continuous mapping of a square  $R = [-a_1, a_1] \times [-a_2, a_2] \subset \mathbb{R}^2$  into  $\mathbb{R}^2$ , the aim of this paper is to develop a bisection method for solving non-linear system of two-dimensional equations  $\mathbf{F}(X) = 0$ ,  $X = (x, y)$  using the above Poincaré-Miranda theorem. We denote by  $\|\cdot\|$  the euclidean norm in  $\mathbb{R}^2$  and we accept  $r$  as a root with tolerance level  $\delta$  if  $\|\mathbf{F}(r)\| \leq \delta$ .

The idea of the algorithm is similar as in the classical one dimensional algorithm, we perform refinement of the domain region in order to preserve the Poincaré-Miranda sign conditions. One of the advantages of this method reside in the fact that it does not require derivatives of  $\mathbf{F}$ , nor the approximation of such derivatives; only the computed values of  $\mathbf{F}$  are required. However, we cannot ensure always its converge.

## 2 The algorithm and its description

This section gives a step-by-step description of our algorithm, the core of it meets in the classical bisection algorithm in one dimension.

**Definition 2.1.** *A quadrisection of a rectangle  $R$  is a refinement into four congruent rectangles  $Q = \{R^1, R^2, R^3, R^4\}$ .*

$R^3$	$R^4$
$R^1$	$R^2$

**Figure 1:** Quadrisection of a rectangle.

We say that a quadrisection  $Q$  verifies the Poincaré-Miranda condition if there exist  $R^i \in Q$  such that  $\mathbf{F} : R^i \rightarrow \mathbb{R}^2$  verifies the condition of Theorem 1.1.

## 2.1 Algorithm procedure

The algorithm proceeds as follows:

1. We start choosing an initial rectangle  $R_0 = [x_{1_0}, x_{2_0}] \times [y_{1_0}, y_{2_0}]$  verifying the Poincaré-Miranda condition on  $\mathbf{F}$ .
2. We locate the center

$$c_1 = \left( \frac{x_{1_0} + x_{2_0}}{2}, \frac{y_{1_0} + y_{2_0}}{2} \right)$$

of  $R_0$ .

3. Generate a first quadrisection  $Q_1$  through  $c_1$ .
4. If  $Q_1$  verifies the Poincaré-Miranda condition, let  $R_1 = [x_{1_1}, x_{2_1}] \times [y_{1_1}, y_{2_1}]$  be the quarter of  $Q_1$  where the conditions of Theorem 1.1 are verified, we chose

$$c_2 = \left( \frac{x_{1_1} + x_{2_1}}{2}, \frac{y_{1_1} + y_{2_1}}{2} \right)$$

the center of  $R_1$ . The recursion is repeated while the Poincaré-Miranda condition are verified, generating a decreasing rectangle sequence  $R_n$ ,

$R_{n+1} \subset R_n = [x_{1_n}, x_{2_n}] \times [y_{1_n}, y_{2_n}]$  where the vertices verify

$$x_{1_0} \leq x_{1_1} \leq x_{1_2} \leq \dots \leq x_{1_n} \leq \dots \leq x_{2_0} \quad (2.1)$$

$$x_{2_0} \geq x_{2_1} \geq x_{2_2} \geq \dots \geq x_{2_n} \geq \dots \geq x_{1_0} \quad (2.2)$$

$$y_{1_0} \leq y_{1_1} \leq y_{1_2} \leq \dots \leq y_{1_n} \leq \dots \leq y_{2_0} \quad (2.3)$$

$$y_{2_0} \geq y_{2_1} \geq y_{2_2} \geq \dots \geq y_{2_n} \geq \dots \geq y_{1_0} \quad (2.4)$$

where the length of the current interval  $[x_{1_n}, x_{2_n}]$ ,  $[y_{1_n}, y_{2_n}]$  is a half of the last iteration,

$$x_{2_n} - x_{1_n} = \frac{x_{2_{n-1}} - x_{1_{n-1}}}{2} = \dots = \frac{x_{2_0} - x_{1_0}}{2^n} \quad (2.5)$$

$$y_{2_n} - y_{1_n} = \frac{y_{2_{n-1}} - y_{1_{n-1}}}{2} = \dots = \frac{y_{2_0} - y_{1_0}}{2^n} \quad (2.6)$$

The root's approximation after  $n$ -th iteration will be,

$$c_n = \left( \frac{x_{1_{n-1}} + x_{2_{n-1}}}{2}, \frac{y_{1_{n-1}} + y_{2_{n-1}}}{2} \right)$$

and the method is stopped until the zero's estimates gives sufficiently accuracy or until the Poincaré-Miranda condition leaves to maintain.

Since we cannot always ensure that a quadrisection of a given rectangle will verify the Poincaré-Miranda condition, we cannot ensure the converge for any map that only has a sign change in a given initial rectangle. So, the accuracy of the root's approximation will depend of the amount of quadrisections that the initial condition allows us. However, if we could built an infinite quadrisection on an initial rectangle the algorithm will converge to a root.

**Theorem 2.2.** *Let  $\mathbf{F} = (f_1, f_2) : R_0 \rightarrow \mathbb{R}^2$  be a continuous map that verifies the Poincaré-Miranda condition and suppose that  $\mathbf{F}$  admits an infinite quadrisection checking the Poincaré-Miranda condition on  $R_0$  then bisection algorithm generates a sequence  $c_n$  such that,*

1.  $c_n \xrightarrow{\|\cdot\|} r$  with  $\mathbf{F}(r) = 0$ .
2.  $\|c_n - r\| \leq \frac{Per(R_0)}{2^{n+1}}$  where  $Per(R_0)$  denotes the perimeter of  $R_0$ .

*Proof.*

1. Since we can assume an infinite quadrisection the sequences 2.1, 2.2, 2.3, 2.4 are monotones and bounded and therefore they converge. From equation 2.5 and 2.6 we have that,

$$\lim_{n \rightarrow \infty} x_{1_n} = \lim_{n \rightarrow \infty} x_{2_n} = r_1 \quad (2.7)$$

$$\lim_{n \rightarrow \infty} y_{1_n} = \lim_{n \rightarrow \infty} y_{2_n} = r_2 \quad (2.8)$$

and from the border conditions,

$$f_1(x_{1_n}, y)f_1(x_{2_n}, y) \leq 0 \quad \text{for all } y \in [y_{1_n}, y_{2_n}] \quad (2.9)$$

$$f_2(x, y_{1_n})f_2(x, y_{2_n}) \leq 0 \quad \text{for all } x \in [x_{1_n}, x_{2_n}] \quad (2.10)$$

Since the diameter of  $R_n$  tends to zero by Cantor's intersection theorem the intersection of the  $R_n$  contains exactly one point,

$$\{p\} = \bigcap_{n=0}^{\infty} R_n$$

and the equations 2.7, 2.8 guarantee that  $p = (r_1, r_2)$ . Then, we can evaluate equations 2.9 and 2.10 in  $p = (r_1, r_2)$  getting,

$$f_1(x_{1_n}, r_2)f_1(x_{2_n}, r_2) \leq 0 \quad \forall n \in \mathbb{N}$$

$$f_2(r_1, y_{1_n})f_2(r_1, y_{2_n}) \leq 0 \quad \forall n \in \mathbb{N}$$

and from the continuity of  $f_1, f_2$  we have that  $f_1(r_1, r_2)^2 \leq 0$ ,  $f_2(r_1, r_2)^2 \leq 0$  getting  $\mathbf{F}(r_1, r_2) = 0$ . On other side it is clear from equations 2.7 and 2.8 that

$$c_n = \left( \frac{x_{1_{n-1}} + x_{2_{n-1}}}{2}, \frac{y_{1_{n-1}} + y_{2_{n-1}}}{2} \right) \xrightarrow{n \rightarrow \infty} (r_1, r_2)$$

2. Let  $c_{1_n}$  and  $c_{2_n}$  be the coordinates of the sequence  $c_n$  we have following estimation

$$|c_{1_n} - r_1| \leq \frac{x_{2_0} - x_{1_0}}{2^n} \quad (2.11)$$

$$|c_{2_n} - r_2| \leq \frac{y_{2_0} - y_{1_0}}{2^n}. \quad (2.12)$$

Indeed, since the sequences  $(x_{1_n})$  and  $(x_{2_n})$  are monotones and bounded by  $r_1$  we get,

$$\begin{aligned} c_{1_n} - r_1 &= \frac{x_{1_{n-1}}}{2} + \frac{x_{2_{n-1}}}{2} - r_1 \leq \frac{x_{1_{n-1}}}{2} + \frac{x_{2_{n-1}}}{2} - x_{1_{n-1}} \\ &= \frac{x_{2_{n-1}}}{2} - \frac{x_{1_{n-1}}}{2} = \frac{x_{2_0} - x_{1_0}}{2^n} \end{aligned}$$

On the other hand,

$$\begin{aligned} c_{1_n} - r_1 &= \frac{x_{1_{n-1}}}{2} + \frac{x_{2_{n-1}}}{2} - r_1 \geq \frac{x_{1_{n-1}}}{2} + \frac{x_{2_{n-1}}}{2} - x_{2_{n-1}} \\ &= -\left(\frac{x_{2_{n-1}}}{2} - \frac{x_{1_{n-1}}}{2}\right) = -\left(\frac{x_{2_0} - x_{1_0}}{2^n}\right) \end{aligned}$$

An analogous computation can be done for  $(y_{1_n})$  and  $(y_{2_n})$  in order to prove the estimation for  $c_{2_n}$ . Joining equality 2.11 and 2.12 we have,

$$\begin{aligned} \|c_n - r\| &= \sqrt{(c_{1_n} - r_1)^2 + (c_{2_n} - r_2)^2} \leq |c_{1_n} - r_1| + |c_{2_n} - r_2| \\ &\leq \frac{x_{2_0} - x_{1_0}}{2^n} + \frac{y_{2_0} - y_{1_0}}{2^n} = \frac{Per(R_0)}{2^{n+1}} \end{aligned}$$

□

The following example shows an infinite application of the bisection algorithm and its convergence to a root.

**Example 2.3.** Consider the map  $\mathbf{F}(x, y) = (y + x - 1, y - e^{-x^2})$ , we start checking the Poincaré-Miranda condition on  $R_0 = [0, 1] \times [0, 1]$ ,

$$\begin{aligned} f_1(0, y) &= y - 1 \leq 0, & f_1(1, y) &= y \geq 0 \\ f_2(x, 0) &= -e^{-x^2} < 0, & f_2(x, 1) &= 1 - e^{-x^2} \geq 0 \end{aligned}$$

then if we considerate the  $R^3$  quarter of each quadrissection, it always will verify the Poincaré-Miranda condition. Let  $x_{1_n} = 0$ ,  $x_{2_n} = \frac{1}{2^n}$ ,  $y_{1_n} = 1 - \frac{1}{2^n}$  and  $y_{2_n} = 1$  be the coordinates of the  $n$ -th  $R^3$  quarter rectangle, we have to prove that

$$f_1(x_{1_n}, y)f_1(x_{2_n}, y) = (y - 1)\left(\frac{1}{2^n} + y - 1\right) \leq 0, \quad 1 - \frac{1}{2^n} \leq y \leq 1.$$

$$f_2(x, y_{1_n})f_2(x, y_{2_n}) = \left(1 - \frac{1}{2^n} - e^{-x^2}\right)(1 - e^{-x^2}) \leq 0, \quad 0 \leq x \leq \frac{1}{2^n}$$

Indeed, the first inequality is clear and follows directly from the domain of  $y$ , the second follows from the fact that the domain of  $x$  implies that

$$1 - \frac{1}{2^n} \leq 1 - x \leq e^{-x^2}$$

getting

$$1 - \frac{1}{2^n} - e^{-x^2} \leq 0, \quad n \in \mathbb{N} \text{ and } 1 - e^{-x^2} \geq 0$$

The  $n$ -th root's approximation is,

$$c_n = \left( \frac{1}{\frac{2^{n-1}}{2}}, \frac{1 - \frac{1}{2^{n-1}} + 1}{2} \right) = \left( \frac{1}{2^n}, 1 - \frac{1}{2^n} \right) \xrightarrow{n \rightarrow \infty} (0, 1)$$

and the error verifies

$$\|c_n - (0, 1)\| = \sqrt{\left(\frac{1}{2^n}\right)^2 + \left(\frac{1}{2^n}\right)^2} = \frac{\sqrt{2}}{2^n} \leq \frac{2}{2^n} = \frac{4}{2^{n+1}} = \frac{Per(R_0)}{2^{n+1}}$$

### 3 Implementation, performance and testing

Throughout this section we will focus in the implementation and performance of the algorithm and we will compare its functionality with other methods on some examples. The bisection algorithm was developed in MATLAB in a set of functions running from a main function. The Poincaré-Miranda conditions on each rectangle were testing in a numerical way looking for the positive and negative sets of the coordinate functions  $f_1(-a_1, \cdot) : [-a_2, a_2] \rightarrow \mathbb{R}$ ,  $f_1(a_1, \cdot) : [-a_2, a_2] \rightarrow \mathbb{R}$ ,  $f_2(\cdot, -a_2) : [-a_1, a_1] \rightarrow \mathbb{R}$  and  $f_2(\cdot, a_2) : [-a_1, a_1] \rightarrow \mathbb{R}$ . In order to clarify the implementation we summarize the algorithm in the following pseudo-code:

---

#### Algorithm 1: Bisection algorithm

---

```

Data:  $R_0$ ,  $F = (f_1, f_2)$ ,  $\delta$ 
Result:  $c$  root's approximation
1 if  $R_0$  verifies P.M. then
2    $c = \text{center of } R_0$ ;
3    $\text{err} = \|F(c)\|$ ;
4    $\text{stop} = 1$ ;
5   while  $(\text{err} > \delta) \wedge (\text{stop} < 3)$  do
6      $[R^1, R^2, R^3, R^4] = \text{Generate quadrisection of } R_0$ ;
7      $\text{sgn}F = \text{check sign of } F \text{ on } R^1$ ;
8      $\text{stop} \leftarrow \text{stop} + 1$ ;
9     if  $\text{sgn}F$  verifies P.M. then
10       $c \leftarrow \text{center of } R^1$ ;
11       $R_0 \leftarrow R^1$ ;
12       $\text{err} \leftarrow \|F(c)\|$ ;
13       $\text{stop} \leftarrow \text{stop} - 1$ ;
14      Pass to next iteration
15     $\text{sgn}F = \text{check sign of } F \text{ on } R^2$ ;
16    if  $\text{sgn}F$  verifies P.M. then
17       $c \leftarrow \text{center of } R^2$ ;
18       $R_0 \leftarrow R^2$ ;
19       $\text{err} \leftarrow \|F(c)\|$ ;
20       $\text{stop} \leftarrow \text{stop} - 1$ ;
21      Pass to next iteration
22     $\text{sgn}F = \text{check sign of } F \text{ on } R^3$ ;
23    if  $\text{sgn}F$  verifies P.M. then
24       $c \leftarrow \text{center of } R^3$ ;
25       $R_0 \leftarrow R^3$ ;
26       $\text{err} \leftarrow \|F(c)\|$ ;
27       $\text{stop} \leftarrow \text{stop} - 1$ ;
28      Pass to next iteration
29     $\text{sgn}F = \text{check sign of } F \text{ on } R^4$ ;
30    if  $\text{sgn}F$  verifies P.M. then
31       $c \leftarrow \text{center of } R^4$ ;
32       $R_0 \leftarrow R^4$ ;
33       $\text{err} \leftarrow \|F(c)\|$ ;
34       $\text{stop} \leftarrow \text{stop} - 1$ ;
35      Pass to next iteration
36 else
37   return Wrong  $R_0$ 

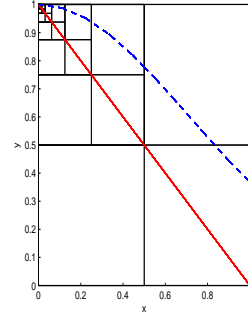
```

---

In order to check the accuracy and performance of the algorithm, we test it throughout different examples. We start testing the algorithm in the system given in the Example 2.3. We took as our starting guess the rectangle  $R_0 = [0, 1] \times [0, 1]$ , in Table 1 we show the behaviour of the sequence throughout different tolerance levels and in Figure 2 we illustrate the procedure for tolerance level  $10^{-15}$ . We have chosen to use 10 digits in the mantissa representation for the root's approximation and its evaluation and 5 digits for the norm evaluation notation.

$\delta$	$c$	$\ \mathbf{F}(c)\ $	steps
1	0.5000000000 0.5000000000	0.2788	1
$10^{-1}$	0.0625000000 0.9375000000	0.0586	4
$10^{-2}$	0.007812500 0.992187500	0.0077	7
$10^{-5}$	0.000007629 0.999992370	7.6293 1e-06	17
$10^{-10}$	5.820799999 1e-11 0.9999999999	5.8207 1e-11	34
$10^{-15}$	0.000000001 1e-06 0.9999999999	8.8817 1e-16	50

**Table 1:** Evolution of root's approximation, norm evaluation and steps performed throughout different tolerance levels.



**Figure 2:** Quadrisection procedure.

In the following steps we test the algorithm in more difficult systems, we will see that although we cannot ensure that there exist an infinite quadrisection the method can be performed in several iterations getting a very good approximation of the root. Let,

$$\mathbf{F}_1(x, y) = (2x - y - e^{-x}, -x + 2y - e^{-y})$$

$$\mathbf{F}_2(x, y) = (\sin(x) + \cos(y) + 2(x - 1), y - 0.5(x - 0.5)^2 - 0.5)$$

$$\mathbf{F}_3(x, y) = (x \cos(y) + y \sin(x) - 0.5, e^{e^{-(x+y)}} - y(1 + x^2))$$

$$\mathbf{F}_4(x, y) = (x^2 - \cos(xy), e^{xy-1} + y)$$

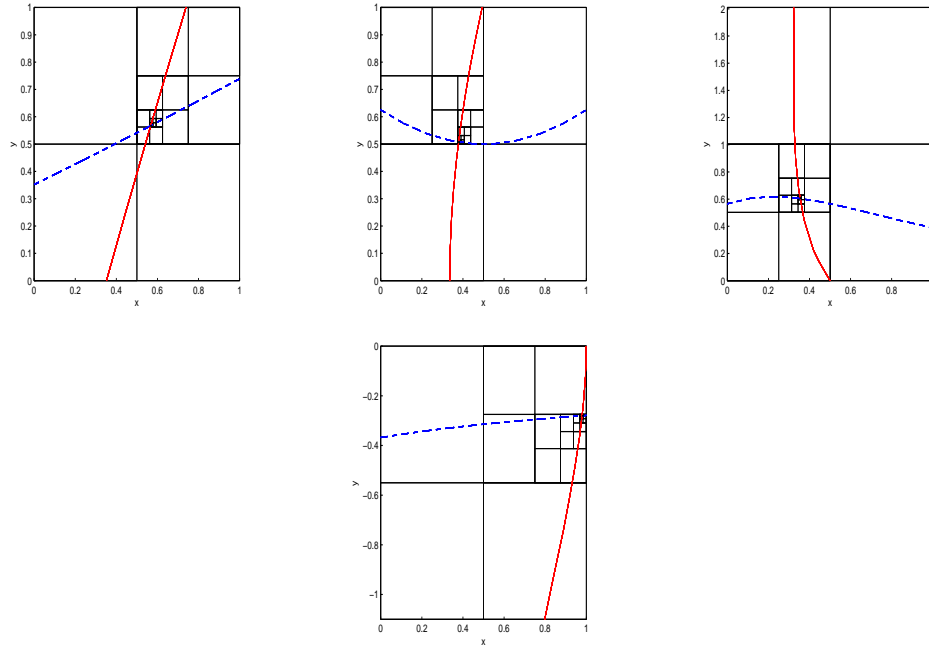
be the testing maps. In Table 2 we show the numerical performance for the testing maps and in Figure 3 we illustrate the algorithm behaviour with the



corresponding quadrisection procedure; in all instances we have setted the tolerance level in  $10^{-15}$ .

$\mathbf{F}$	$c$	$\mathbf{F}(c)$	$\ \mathbf{F}(c)\ $	$R_0$	steps
$\mathbf{F}_1$	0.567143290	0.111022302 1e-15	1.5700 1e-16	$[0, 1]$	50
	0.567143290	0.111022302 1e-15		$\times [0, 1]$	
$\mathbf{F}_2$	0.378316879	0.289874793 1e-06	1.0132 1e-06	$[0, 1]$	21
	0.507402420	-0.970891051 1e-06		$\times [0, 1]$	
$\mathbf{F}_3$	0.353246245	-0.513910910 1e-06	5.1466 1e-07	$[0, 1.001]$	23
	0.606081776	0.027887973 1e-06		$\times [0, 2.01]$	
$\mathbf{F}_4$	0.981124222	-0.081975217 1e-07	3.0348 1e-08	$[0, 1]$	26
	-0.279615980	0.292207502 1e-07		$\times [-1.1, 0]$	

**Table 2:** Root's approximation, evaluation, norm evaluation, starting rectangle and steps performed.



**Figure 3:** The first row illustrates the algorithm procedure for  $\mathbf{F}_1, \mathbf{F}_2$  and the second for  $\mathbf{F}_3, \mathbf{F}_4$ . The solid red line represents the first coordinate and the dashed blue line the second.

To conclude our analysis, we are going to compare our root's approximation

with the solver toolbox approximation. The MATLAB's toolbox attempts to solve systems of equations by minimizing the sum of squares of the components, if the sum of squares is close to zero, the system of equation is solved. In Table 3 we illustrate our comparison; it can be seen that the relative error, taken respect to the solution of the toolbox in the euclidean norm, is very small showing a high performance of the method in these cases. On other hand, the computing time of the bisection algorithm is greater than the computing time getting by the solver algorithm; in Table 4 we show the time performance of both methods.

<b>F</b>	<b><math>c</math></b>	<b>Toolbox solver</b>	<b>F(Toolbox solver)</b>	<b>Rel. error</b>
<b>F<sub>1</sub></b>	0.567143290	0.567143285	-0.692733614 1e-08	8.8161 1e-09
	0.567143290	0.567143285	-0.692733614 1e-08	
<b>F<sub>2</sub></b>	0.378316879	0.378316940	-0.137471811 1e-09	1.5245 1e-06
	0.507402420	0.507403383	-0.074302342 1e-09	
<b>F<sub>3</sub></b>	0.353246245	0.353246568	-0.725604261 1e-07	4.6706 1e-07
	0.606081776	0.606081721	0.215374862 1e-07	
<b>F<sub>4</sub></b>	0.981124222	0.981124255	0.639003346 1e-07	3.4766 1e-08
	-0.279615980	-0.279615993	0.097238697 1e-07	

**Table 3:** Comparison between bisection algorithm and solver MATLAB toolbox.

<b>F</b>	<b>Bisection elapsed time</b>	<b>Solver elapsed time</b>
<b>F<sub>1</sub></b>	1.4425	0.0153
<b>F<sub>2</sub></b>	0.5512	0.0245
<b>F<sub>3</sub></b>	0.5789	0.0617
<b>F<sub>4</sub></b>	0.8718	0.0224

**Table 4:** Time performance comparison between bisection algorithm and solver MATLAB toolbox. The values are expressed in seconds.

## 4 Conclusion

In this work we have developed an original two-dimensional root-finding bisection method using a generalization of Bolzano theorem. Since we cannot always guarantee that a quadrisection will verify the Poincaré-Miranda condition we cannot always ensure the converge of the method, however for several

systems it has very good performance in the root's approximation. Moreover, we have proved that the method converges to a root for systems that allow infinite quadrisections. The accuracy of the method depends of the number of quadrisections allowed by the initial guess, so it plays a key role. On other side the main difference of this method respect to the classical Newton and optimization algorithms reside in the fact that it does not require derivatives of  $\mathbf{F}$  avoiding the problem of ill-conditioned Jacobian matrix.

## References

- [1] W. Kulpa, The Poincaré-Miranda theorem, Amer. Math. Monthly 104 (1997), no. 6, 2513-2530.
- [2] D. H. Lehmer, (April 1961), A Machine Method for Solving Polynomial Equations, Journal of the ACM, 8 (2): 151-162.
- [3] C. Miranda, Un'osservazione su un teorema di Brouwer, Boll. Un. Mat. Ital. (2) 3 (1940), 5-7.
- [4] H. Poincaré, Sur certaines solutions particulières du problème des trois corps, C. R. Acad Sci.Paris 97 (1883), 251-252 (French).
- [5] H. Poincaré, Sur certaines solutions particulières du problème des trois corps, Bulletin Astronomique 1 (1884), 65-74 (French).
- [6] N. Rouche and J. Mawhin, Équations Différentielles Ordinaires. Tome I: Théorie Générale, Mason at Cie, Éditeurs, Paris, 1973.
- [7] Herbert S. Wilf (1978), A Global Bisection Algorithm for Computing the Zeros of Polynomials in the Complex Plane, Journal of the ACM, 25 (3).

Manuel López Galván.

Instituto Argentino de Matemática (CONICET) y Universidad de Buenos Aires, Argentina.

e-mail: mlopezgalvan@hotmail.com